

AUTOMATIC LOG-IN/LOG-OUT COMPUTER SYSTEM

Patent Number: JP9251331

Publication date: 1997-09-22

Inventor(s): SASAKI MASATOMI

Applicant(s): HITACHI LTD

Requested Patent: JP9251331

Application Number: JP19960060588 19960318

Priority Number(s):

IPC Classification: G06F1/26; G06F15/00; G06K19/00

EC Classification:

Equivalents:

Abstract

PROBLEM TO BE SOLVED: To reduce a trouble for saving the power of a system and for security protection by executing log-in and log-out at the same time as power on/off by inserting/extracting an ID card and the other cards.

SOLUTION: When the ID card 30 is inserted into the ID card insertion port 21 of a card reader 20, a power switch 22 is turned on and a computer system 10 is started. Individual information of the ID card 30 is read and ID is judged whether the content is previously registered in a computer system main body 11 or not. When ID is not registered, power is turned off. When it is registered, log-in is executed and an operator can input data from a keyboard 13 and he can execute work while he views a display device 12. For terminating of interrupting work, the power switch 22 senses the extraction of the ID card 30 when it is extracted, the computer system 10 saves a file during editing and power is turned off after a log-out processing.

Data supplied from the **esp@cenet** database - I2

(19) 日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平9-251331

(43) 公開日 平成9年(1997)9月22日

(51) Int.Cl.⁶
G 0 6 F 1/26
15/00
G 0 6 K 19/00

識別記号
3 3 0

序内整理番号

F I
G 0 6 F 1/00
15/00
G 0 6 K 19/00

技術表示箇所
3 3 4 A
3 3 0 B
T

審査請求 未請求 請求項の数1 O L (全3頁)

(21) 出願番号 特願平8-60588

(22) 出願日 平成8年(1996)3月18日

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72) 発明者 佐々木 正富

愛知県尾張旭市晴丘町池上1番地株式会社
日立製作所オフィスシステム事業部内

(74) 代理人 弁理士 小川 勝男

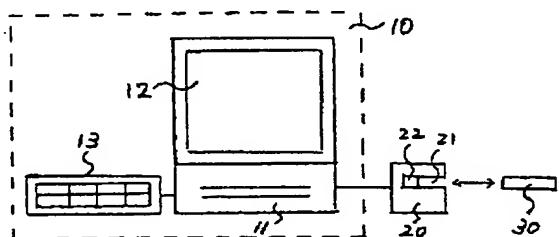
(54) 【発明の名称】 自動ログイン・ログアウト計算機システム

(57) 【要約】

【課題】 計算機システムの省電力化とセキュリティ保護に対する、オペレータの電源ON・OFF並びにログイン・ログアウトの手間を軽減する。

【解決手段】 図1のように、計算機システム10に電源スイッチ22を内蔵したカードリーダ20を取り付け。これにより、IDカード30の抜き差しによって計算機システム10の電源をON・OFFすると共に個人情報を読み込み、オペレータの自動ログイン・ログアウトが可能となる。

図1



【特許請求の範囲】

【請求項1】 IDカードおよびその他カード類の抜き差しによって、電源をON・OFFすると同時に、ログイン・ログアウトを自動的に行なえる計算機システム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、計算機システムの省電力化と、オペレータ操作の手間軽減に関する。

【0002】

【従来の技術】 従来の技術は、ログイン・ログアウトをオペレータがキーボード入力することにより行なうこと一般である。また、特開昭60-225964号に記載のように、オペレータはログイン前に計算機システムの電源をあらかじめONにしておく必要があった。

【0003】

【発明が解決しようとする課題】 前記従来技術では、オペレータは席を外す度にセキュリティ保護のため、計算機システムのログイン・ログアウト操作をキーボード入力により行なっており、手間がかかった。また、ログインする前の計算機システムの電源ONや、ログアウトした後の省電力のための電源OFFの手間が必要であり、問題があった。

【0004】 本発明は、上記のような問題を解決するために考案したものであって、その目的は計算機システムの省電力化を図るとともに、セキュリティ保護に対するオペレータの手間を軽減することである。

【0005】

【課題を解決するための手段】 上記目的を達成するため、計算機システムにカードリーダを取り付け、カードリーダにはIDカードの抜き差しによって計算機システムの電源をON・OFFするスイッチを取り付ける。

【0006】

【発明の実施の形態】 以下、本発明の一実施例を図1、図2により説明する。図1は、本発明の特徴を有するシステム構成図を示す。図2は、図1のシステムの動作を示す流れ図である。

【0007】 図1において、10は計算機システム、11は計算機システム本体、12は表示装置、13はキーボード、14はカードリーダ、15はIDカード挿入口、16は電源スイッチ、17はIDカードである。

ボード、20はカードリーダ、21はIDカード挿入口、22は電源スイッチ、30はIDカードである。

【0008】 次に、図2の流れ図に基づいて、図1のシステムの動作を説明する。

【0009】 まず、図1に示すIDカード30をカードリーダ20のIDカード挿入口21に挿入すると(ステップ100)、電源スイッチ22がONされ(ステップ102)、計算機システム10が起動する(ステップ104)。続いて、IDカード30の個人情報がカードリ

1- ボード20によって読み込まれ、その内容があらかじめ計算機システム本体11に登録されているIDか否かを判定し(ステップ106)、登録されていないIDであれば電源OFFとなり(ステップ118)、登録されればログインされ(ステップ108)、オペレータはキーボード13からの入力が可能となり、表示装置12を見ながら作業が行なえる(ステップ110)。次に、オペレータが作業を終了、又は中断する場合には、IDカード30を引き抜くと(ステップ112)、電源スイッチ22がIDカード30が引き抜かれたことを感知し、20 計算機システム10に作業終了の信号を送り、それを受けた計算機システム10は、自動的にオペレータ編集中のファイル等をセーブし(ステップ114)、ログアウト処理を行なった後(ステップ116)、電源をOFFする(ステップ118)。

【0010】

【発明の効果】 本発明により、計算機システムの省電力化とセキュリティ保護に対する、オペレータの電源ON・OFF並びにログイン・ログアウトの手間を軽減することができる。

【図面の簡単な説明】

【図1】 本発明のシステム構成図である。

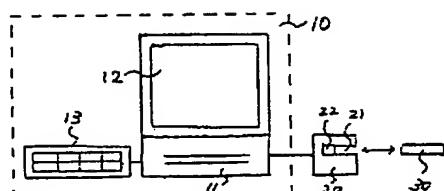
【図2】 本システムの動作を示す流れ図である。

【符号の説明】

10…計算機システム、11…計算機システム本体、12…表示装置、13…キーボード、14…カードリーダ、15…IDカード挿入口、16…電源スイッチ、30…IDカード。

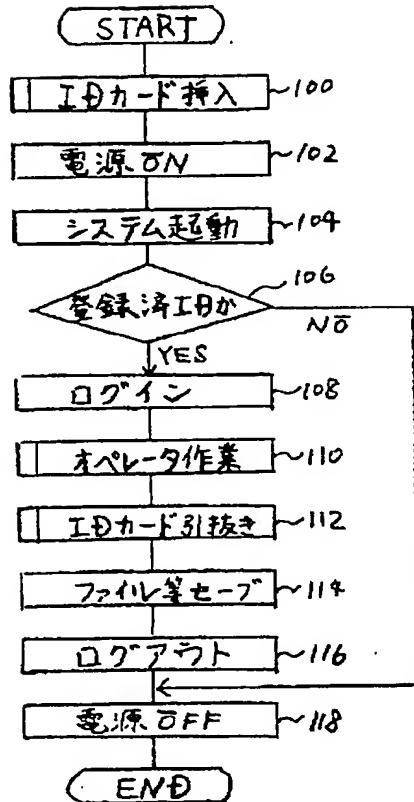
【図1】

図1



【図2】

②2.



図中 [] はオペレータ
が行ない、[] は
システムが自動的に
行なう。

DEVICE AND METHOD FOR MANAGING NETWORK

Patent Number: JP2000353142

Publication date: 2000-12-19

Inventor(s): HAMADA NOBORU

Applicant(s): CANON INC

Requested Patent: JP2000353142

Application Number: JP19990165570 19990611

Priority Number(s):

IPC Classification: G06F13/00; G06F13/14

EC Classification:

Equivalents:

Abstract

PROBLEM TO BE SOLVED: To reduce traffic for searching a network device.

SOLUTION: In the case of searching a network device, a packet for search is executed once at the time of starting the search (S1301), the research of each device searched at the time of starting is repeated one by one at a prescribed interval (S1305). Consequently a situation where network traffic increases can be prevented by broadcasting the packet for search. Since the broadcast packet is periodically transmitted, the broadcast packet is successively transmitted to devices registered in a device table instead of updating the device table and only whether each device responds to the packet or not can be checked. Consequently the device table can be updated in a comparatively small network traffic.

Data supplied from the **esp@cenet** database - I2

(19)日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11)特許出願公開番号
特開2000-353142
(P2000-353142A)

(43)公開日 平成12年12月19日 (2000.12.19)

(51)Int.Cl'	識別記号	F I	マークド(参考)
G 06 F 13/00	3 5 7	G 06 F 13/00	3 5 7 A 5 B 0 1 4
	3 5 3		3 5 3 B 5 B 0 8 9
13/14	3 3 0	13/14	3 3 0 A

審査請求 未請求 請求項の数6 O L (全20頁)

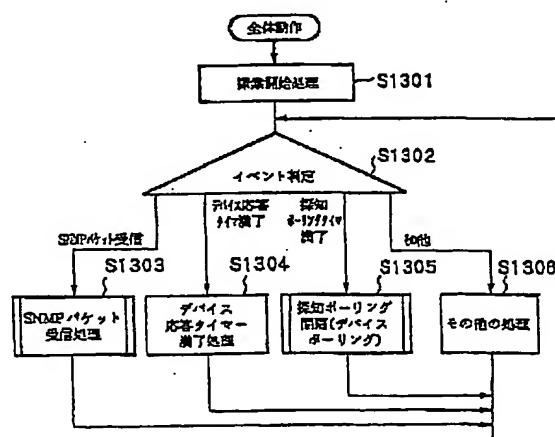
(21)出願番号 特願平11-165570
(22)出願日 平成11年6月11日(1999.6.11)

(71)出願人 000001007
キヤノン株式会社
東京都大田区下丸子3丁目30番2号
(72)発明者 浜田 昇
東京都大田区下丸子3丁目30番2号 キヤ
ノン株式会社内
(74)代理人 100076428
弁理士 大塚 康徳 (外2名)
Fターム(参考) 5B014 FB04 HC02
5B089 GB02 HA06 JA35 JB15 KA13
KB04 KC20 KC30 KC44

(54)【発明の名称】 ネットワーク管理装置及び方法

(57)【要約】

【課題】ネットワークデバイスの探索のためのトラフィックを減らす。
【解決手段】ネットワークデバイスの探索時には、起動時に探知のためのパケットを一度行った後(S1301)、所定の間隔で、起動時に探知したデバイスに対して、一度につづつ再探知を繰り返す(S1305)。こうして、探知のためにパケットをブロードキャストしてネットワークトラフィックをあげてしまう事態を防止する。



【特許請求の範囲】

【請求項1】 ブロードキャストパケットによってネットワークに接続されているデバイスを検出するためのネットワークデバイス探知手段と、前記ネットワークデバイス探知手段により検出されたデバイスを登録するためのデバイステーブルと、前記デバイステーブルに登録されたデバイスに対して、所定間隔で順次探知シーケンスを遂行する個別デバイス探知手段と、前記個別デバイス探知手段により検出された情報により、前記デバイステーブルを更新するデバイステーブル更新手段とを備えることを特徴とするネットワークデバイス管理装置。

【請求項2】 前記デバイステーブルの登録内容の出力を指示された際に、前記デバイステーブルに登録された各デバイスに対して探知シーケンスを遂行し、検出された情報により前記デバイステーブルを更新する手段と、前記デバイステーブルを出力する出力手段とを更に備えることを特徴とするネットワークデバイス管理装置。

【請求項3】 ブロードキャストパケットによってネットワークに接続されているデバイスを検出し、検出されたデバイスをデバイステーブルに登録するためのネットワークデバイス探知工程と、前記デバイステーブルに登録されたデバイスに対して、所定間隔で順次探知シーケンスを遂行する個別デバイス探知工程と、

前記個別デバイス探知工程により検出された情報により、前記デバイステーブルを更新するデバイステーブル更新工程とを備えることを特徴とするネットワークデバイス管理方法。

【請求項4】 前記デバイステーブルの登録内容の出力を指示された際に、前記デバイステーブルに登録された各デバイスに対して探知シーケンスを遂行し、検出された情報により前記デバイステーブルを更新する工程と、前記デバイステーブルを出力する出力工程とを更に備えることを特徴とするネットワークデバイス管理方法。

【請求項5】 コンピュータにより、ブロードキャストパケットによってネットワークに接続されているデバイスを検出するためのネットワークデバイス探知手段と、前記ネットワークデバイス探知手段により検出されたデバイスを登録するためのデバイステーブルと、前記デバイステーブルに登録されたデバイスに対して、所定間隔で順次探知シーケンスを遂行する個別デバイス探知手段と、前記個別デバイス探知手段により検出された情報により、前記デバイステーブルを更新するデバイステーブル更新手段とを実現するためのコンピュータプログラムを格納するコンピュータ可読メモリ。

【請求項6】 前記デバイステーブルの登録内容の出力

を指示された際に、前記デバイステーブルに登録された各デバイスに対して探知シーケンスを遂行し、検出された情報により前記デバイステーブルを更新する手段と、前記デバイステーブルを出力する出力手段とを更に実現するためのコンピュータプログラムを格納する請求項5に記載のコンピュータ可読メモリ。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、例えばコンピュータネットワークに接続されたデバイス等を管理するためのネットワークデバイス管理装置及び方法に関する。

【0002】

【従来の技術】 近年、コンピュータを相互に接続したローカルエリアネットワーク（LAN）が普及しており、このようなローカルエリアネットワークは、ビルのフロアまたはビル全体、ビル群（構内）、地域、あるいはさらに大きいエリアにわたって構築することができる。このようなネットワークは更に相互接続され、世界的規模のネットワークにも接続することができる。このような相互接続されたそれぞれのLANは、多様なハードウェア相互接続技術といつものネットワークプロトコルを持つことがある。

【0003】 他と切り離された簡単なLANは個々のユーザが管理することができる。すなわち、ユーザが機器を取り替えたり、ソフトウェアをインストールしたり、問題点を診断したりすることができる。

【0004】 しかし一方、規模の大きい複雑なLANや相互接続された大きなLANグループは「管理」を必要とする。「管理」とは、人間のネットワーク管理者とその管理者が使用するソフトウェアの両方による管理を意味する。本願においては、「管理」とはシステム全体を管理するためのソフトウェアによる管理を意味し、「ユーザ」とはネットワーク管理ソフトウェアを使用する人を意味するものとする。このユーザは、通常、システム管理責任者である。ユーザは、ネットワーク管理ソフトウェアを使うことによって、ネットワーク上で管理データを得て、このデータを変更することができる。

【0005】 大規模ネットワークシステムは、通常、機器の増設と除去、ソフトウェアの更新、及び問題の検出などを絶えず行うことが必要な動的システムである。一般に、様々な人が所有する、様々な業者から供給される様々なシステムが存在する。

【0006】 このような大規模ネットワークシステムを構成するネットワーク上のデバイスを管理するための方法として、これまでにいくつかの試みが数多くの標準機関でなされている。国際標準化機構（ISO）は開放型システム間相互接続（Open System Interconnection, OSI）モデルと呼ばれる汎用基準フレームワークを提供した。ネットワーク管理プロトコルのOSIモデルは、共通管理情報プロトコル（Common Management Information Protocol, CMIP）である。

3

mation Protocol, CMIP) と呼ばれる。CMIPはヨーロッパの共通ネットワーク管理プロトコルである。
 【0007】また近年では、より共通性の高いネットワーク管理プロトコルとして、簡易ネットワーク管理プロトコル (Simple Network Management Protocol, SNMP) と呼ばれるCMIPに関連する一変種のプロトコルがある。(「TCP/IPネットワーク管理入門実用的な管理をめざして」M.T.ローズ=著／西田竹志=訳(株)トッパン発行1992年8月20日初版を参照。) このSNMPネットワーク管理技術によれば、ネットワーク管理システムには、少なくとも1つのネットワーク管理ステーション (NMS) 、各々がエージェントを含むいくつかの管理対象ノード、及び管理ステーションやエージェントが管理情報を交換するために使用するネットワーク管理プロトコルが含まれる。ユーザは、NMS上でネットワーク管理ソフトウェアを用いて管理対象ノード上のエージェントソフトウェアと通信することにより、ネットワーク上のデータを得、またデータを変更することができる。

【0008】ここでエージェントとは、各々のターゲット装置についてのバックラウンドプロセスとして走るソフトウェアである。ユーザがネットワーク上の装置に対して管理データを要求すると、管理ソフトウェアはオブジェクト識別情報を管理パケットまたはフレームに入れてターゲットエージェントへ送り出す。エージェントは、そのオブジェクト識別情報を解釈して、そのデータを取り出し、そのデータをパケットに入れてユーザに送り返す。時には、データを取り出すために対応するプロセスが呼び出される場合もある。

【0009】またエージェントは、自分の状態に関するデータをデータベースの形式で保持している。このデータベースのことを、MIB (Management Information Base) と呼ぶ。図4は、MIBの構造を示す概念図である。図4に示すように、MIBは木構造のデータ構造をしており、全てのノードが一意に番号付けされている。図4において、かっこ内に書かれている番号が、そのノードの識別子である。例えば、図4においてノード401の識別子は1である。ノード402の識別子は、ノード401の下の3なので、1・3と表記される。同様にして、ノード403の識別子は、1・3・6・1・2と表記される。このノードの識別子のことを、オブジェクト識別子(OBJECT IDENTIFIER)と呼ぶ。

【0010】このMIBの構造は、管理情報構造 (SMI : Structure of Management Information) と呼ばれ、RFC1155 Structure and Identification of Management Information for TCP/IP-based Internetsで規定されている。

【0011】図4には、標準として規定されているMIBのうち、一部のもののみを抜き出して記載してある。

【0012】次に、SNMPプロトコルについて簡単に説明する。ネットワーク管理ユーティリティソフトウェアが動作しているPC(以下、マネージャと呼称する)とSNMPエージェントが動作している管理対象ネットワークデバイス(以下、エージェントと呼称する)とは、SNMPプロトコルを用いて通信を行う。SNMPプロトコルには5種類のコマンドがあり、それぞれGet-request、Get-next-request、Get-response、Set-request、Trapと呼ばれる。これらのコマンドがマネージャとエージェントの間でやりとりされる様子を図8に示す。

【0013】Get-requestおよびGet-next-requestは、マネージャがエージェントのMIBオブジェクトの値を取得するため、マネージャがエージェントに対して送出するコマンドである。このコマンドを受け取ったエージェントは、MIBの値をマネージャに通知するために、マネージャに対してGet-responseコマンドを送出する(801および802)。

【0014】Set-requestは、マネージャがエージェントのMIBオブジェクトの値を設定するため、マネージャがエージェントに対して送出するコマンドである。このコマンドを受け取ったエージェントは、設定結果をマネージャに通知するために、マネージャに対してGet-responseコマンドを送出する(803)。

【0015】Trapは、エージェントが自分自身の状態の変化をマネージャに対して通知するために、エージェントがマネージャに対して送出するコマンドである(804)。

【0016】図7に、Trap以外のコマンド、即ちGet-request、Get-next-request、Get-responseおよびSet-requestのフォーマットを示す。SNMPメッセージは、バージョン701、コミュニティ名702およびPDUと呼ばれる領域703からなる。PDU703を詳細に示したものが710である。PDU710は、PDUタイプ711、リクエストID712、エラーステータス713、エラーインデックス714およびMIB情報715からなる。PDUタイプ711には、コマンドを識別する値が格納される。即ちこのフィールドの値が0であるならばGet-request、1であるならばGet-next-request、2であるならばGet-response、3であるならばSet-requestと識別される。また、エラーステータス713には、エラー情報を示す値が格納される。エラーがない場合には、このフィールドの値は0である。また、MIB情報715には、オブジェク

40
 40
 40
 40
 50

トIDとその値が組みになって格納される。

【0018】次に、管理が必要な大規模なネットワークについて説明する。

【0019】図1は、プリンタをネットワークに接続するためのネットワークボード(NB)101を、開放型アーキテクチャを持つプリンター102へつなげた場合を示す図である。NB101はローカルエリアネットワーク(LAN)100へ、例えば、同軸コネクタをもつEthernetインターフェース10Base-2や、RJ-45を持つ10Base-T等のLANインターフェースを介してつながれている。

【0020】PC103やPC104等の複数のパソコンコンピューター(PC)もまた、100に接続されており、ネットワークオペレーティングシステムの制御の下、これらのPCはNB101と通信することができる。PCの一つ、例えばPC103を、ネットワーク管理部として使用するように指定することができる。PCに、PC104に接続されているプリンター105のようなプリンタを接続してもよい。

【0021】また、LAN100にファイルサーバー106が接続されており、これは大容量(例えば100億バイト)のネットワークディスク107に記憶されたファイルへのアクセスを管理する。プリントサーバー108は、接続されたプリンター109a及び109b、又は遠隔地にあるプリンター105などのプリンタに印刷を行わせる。また他の図示しない周辺機器をLAN100に接続してもよい。

【0022】更に詳しくは、図1に示すネットワークは、様々なネットワーカメンバー間で効率良く通信を行うために、NovellやUNIXのソフトウェアなどのネットワークソフトウェアを使用することができる。どのネットワークソフトウェアを使用することも可能であるが、例えば、Novell社のNetWare(NetWareの商標。以下省略)ソフトウェアを使用することができる。このソフトウェアパッケージに関する詳細な説明は、NetWareパッケージに同梱されているオンラインドキュメントーションを参照のこと。これは、Novell社からNetWareパッケージとともに購入可能である。

【0023】図1の構成について簡単に説明すると、ファイルサーバー106は、LANメンバー間でデータのファイルの受信や、記憶、キューイング、キャッシング、及び送信を行う、ファイル管理部としての役割を果たす。例えば、PC103及びPC104それぞれによって作られたデータファイルは、ファイルサーバー106へ送られ、ファイルサーバー106はこれらのデータファイルを順に並べ、そしてプリントサーバー108からのコマンドに従って、並べられたデータファイルをプリンター109aへ送信する。

【0024】またPC103とPC104はそれぞれ、データファイルの生成や、生成したデータファイルのL

AN100への送信や、また、LAN100からのファイルの受信や、更にそのようなファイルの表示及び/又は処理を行うことのできる、通常のPCで構成される。図1にパソコンコンピューター機器が示されているが、ネットワークソフトウェアを実行するのに適切であるような、他のコンピューター機器を含んでもよい。例えば、UNIXのソフトウェアを使用している場合に、UNIXワークステーションをネットワークに含んでもよく、これらのワークステーションは、適切な状況下で、図示されているPCと共に使用することができる。

【0025】通常、LAN100などのLANは、一つの建物内の一階又は連続した複数の階でのユーザーグループ等の、幾分ローカルなユーザーグループにサービスを提供する。例えば、ユーザーが他の建物や他県に居るなど、あるユーザーが他のユーザーから離れるに従って、ワイドエリアネットワーク(WAN)を作つてもよい。WANは、基本的には、いくつかのLANを高速度サービス総合デジタルネットワーク(ISDN)電話線等の高速度デジタルラインで接続して形成された集合体である。従つて、図1に示すように、LAN100と、LAN110と、LAN120とは変調/復調(MODEM)ノードンスponder130及びバックボーン140を介して接続されWANを形成する。これらの接続は、数本のバスによる単純な電気的接続である。それぞれのLANは専用のPCを含み、また、必ずしも必要なわけではないが、通常はファイルサーバー及びプリントサーバーを含む。

【0026】従つて図1に示すように、LAN110は、PC111と、PC112と、ファイルサーバー113と、ネットワークディスク114と、プリントサーバー115と、プリンター116及びプリンター117とを含む。対照的に、LAN120はPC121とPC122のみを含む。LAN100と、LAN110と、LAN120とに接続されている機器は、WAN接続を介して、他のLANの機器の機能にアクセスすることができる。

【0027】エージェントの実装例として、プリンタをネットワークに接続するためのネットワークボード上にエージェントを実装することが考えられる。これにより、プリンタをネットワーク管理ソフトウェアによる管理の対象とすることができます。ユーザは、ネットワーク管理ソフトウェアを用いて制御対象のプリンタの情報を得、また状態を変更することができる。より具体的には、例えばプリンタの液晶ディスプレイに表示されている文字列を取得したり、デフォルトの給紙カセットを変更したりすることができる。以下、エージェントを実装したネットワークボード(NB)をプリンタに接続する実施形態について説明する。図2に示すように、好ましくは、NB101は、プリンター102の内部拡張1/0スロットに内蔵されており、NB101は、下に示す

7
処理及びデータ記憶機能を持つ「埋め込まれた」ネットワークノードとなる。このNB101の構成により、大きなマルチエリアWANネットワークを統括及び管理するための、特徴的な補助機能を持つという利点をもたらす。これらの補助機能は、例えば、ネットワーク上の遠隔地（ネットワーク統括者の事務所など）からのプリンタ制御及び状態観察や、各印刷ジョブ後の次のユーザーのための保証初期環境を提供するためのプリンター構成の自動管理、及びプリンターの負荷量を特徴付け、あるいはトナーカートリッジの交換スケジュールを組むためにネットワークを通してアクセスできる、プリンターログ又は使用統計を含む。

【0028】このNB設計において重要な要因は、共有メモリ等の両方向インターフェースを介して、NB101からプリンタ制御状態にアクセスする機能である。共有メモリ以外に、SCSIインターフェース等のインターフェースを使用することもできる。これにより、多数の便利な補助機能のプログラムができるよう、プリンタ操作情報をNB101又は外部ネットワークノードへ送出することができる。印刷画像データ及び制御情報のブロックは、NB101上にあるマイクロプロセッサーによって構成され、共有メモリに記述され、そして、プリンタ102によって読み込まれる。同様に、プリンタ状態情報は、プリンタ102から共有メモリへ送られ、そこからNBプロセッサーによって読み込まれる。

【0029】図2は、NB101をプリンタ102にインストールした状態を示す断面図である。図2に示すように、NB101はネットワーク接続の為のフェースプレート101bを設置した印刷回路ボード101aから構成されており、コネクタ170を介してプリンタインターフェースカード150に接続されている。プリンタインターフェースカード150は、プリンタ102のプリンタエンジンを直接制御する。印刷データ及びプリンタ状態コマンドは、NB101からコネクタ170を介して、プリンタインターフェースカード150へ入力され、また、プリンタ状態情報はプリンタインターフェースカード150からやはりコネクタ170を介して得られる。NB101はこの情報を、フェースプレート101bのネットワークコネクタを介して、LAN100上で通信する。同時に、プリンタ102は、従来のシリアルポート102a及びパラレルポート102bから、印刷データを受信することもできる。

【0030】図3は、NB101とプリンタ102とLAN100との電気的接続を示すブロック図である。NB101は、LAN100へはLANインターフェースを介して、プリンタ102へはプリンタインターフェースカード150を介して直接接続されている。NB101上にはNB101を制御するためのマイクロプロ

ロセッサー301と、マイクロプロセッサー301の動作プログラムを格納するためのROM303と、マイクロプロセッサー301がプログラムを実行する上でワークとして用いるためのRAM302と、NB101とプリンタインターフェースカード150とが相互にデータをやりとりするための共有メモリ200があり、内部バスを通じて相互に接続されている。NB101がSNMPのエージェントとして動作するためのプログラムはROM303に格納されている。マイクロプロセッサー301は、ROM303に格納されたプログラムに従って動作し、ワークエリアとしてRAM302を用いる。また、プリンタインターフェースカード150と相互に通信するためのバッファ領域として共有メモリ200を用いる。

【0031】プリンタインターフェースカード150上のマイクロプロセッサー151はNB101とのデータのアクセスを、NB101に設置されている共有メモリ200を介して行う。プリンタインターフェースカード150上のマイクロプロセッサー151は、実際に印刷機構を動かすプリンタエンジン160とも通信する。

【0032】一方、ネットワーク管理ソフトウェアが稼動するPC側について、以下で説明する。

【0033】図5は、ネットワーク管理ソフトウェアが稼動可能なPCの構成を示すブロック図である。

【0034】図5において、500は、ネットワーク管理ソフトウェアが稼動するPCであり、図1における103と同等である。PC500は、ROM502もしくはハードディスク(HD)511に記憶された、あるいはフロッピーディスクドライブ(FD)512より供給されるネットワーク管理プログラムを実行するCPU501を備え、システムバス504に接続される各デバイスを総括的に制御する。

【0035】503はRAMで、CPU501の主メモリ、ワークエリア等として機能する。

【0036】505はキーボードコントローラ(KB-C)で、キーボード(KB)509や不図示のポイントティングデバイス等からの指示入力を制御する。506はCRTコントローラ(CRTC)で、CRTディスプレイ(CRT)510の表示を制御する。507はディスクコントローラ(DKC)で、ブートプログラム、種々のアプリケーション、編集ファイル、ユーザファイルそしてネットワーク管理プログラム等を記憶するハードディスク(HD)511およびフロッピーディスクコントローラ(FD)512とのアクセスを制御する。508はネットワークインターフェースカード(NIC)で、LAN100を介して、エージェントあるいはネットワーク機器と双方向にデータをやりとりする。

【0037】次に、従来例におけるネットワーク管理ソフトウェアの構成について説明する。

【0038】従来例におけるネットワーク管理装置は、

図5に示したようなネットワーク管理装置を実現可能なPCと同様の構成のPC上に実現される。ハードディスク(HD)511には、後述のすべての説明で動作主体となる本願に係るネットワーク管理ソフトウェアのプログラムが格納される。後述のすべての説明において、特に断りのない限り、実行の主体はハード上はCPU501である。一方、ソフトウェア上の制御の主体は、ハードディスク(HD)511に格納されたネットワーク管理ソフトウェアである。本従来例においては、OSは例えば、ウインドウズ95(マイクロソフト社製)を想定しているが、これに限るものではない。

【0039】なお本願に係るネットワーク管理プログラムは、フロッピーディスクやCD-ROMなどの記憶媒体に格納された形で供給されても良く、その場合には図5に示すフロッピーディスクコントローラ(FD)512または不図示のCD-ROMドライブなどによって記憶媒体からプログラムが読み取られ、ハードディスク(HD)511にインストールされる。

【0040】図6は、本従来例に係るネットワーク管理ソフトウェアのモジュール構成図である。このネットワーク管理ソフトウェアは、図5におけるハードディスク511に格納されており、CPU501によって実行される。その際、CPU501はワークエリアとしてRAM503を使用する。

【0041】図6において、601はデバイスリストモジュールと呼ばれ、ネットワークに接続されたデバイスを一覧にして表示するモジュールである。(一覧表示の様子については、後ほど図15を用いて説明する。) 602は全体制御モジュールと呼ばれ、デバイスリストからの指示をもとに、他のモジュールを統括する。603はコンフィグレータと呼ばれ、エージェントのネットワーク設定に関する特別な処理を行うモジュールである。604は、探索モジュールと呼ばれ、ネットワークに接続されているデバイスを探索するモジュールである。探索モジュール604によって探索されたデバイスが、デバイスリスト601によって一覧表示される。605は、プリントジョブの状況をNetWare API616を用いてネットワークサーバから取得するNetWareジョブモジュールである。(なお、NetWare APIについては、例えばNovell社から発行されている"NetWare Programmer's Guide for C"等を参照のこと。この書籍はノベル株式会社から購入可能である。) 606および607は後述するデバイス詳細ウィンドウを表示するためのUI(User Interface)モジュールであり、詳細情報を表示する対象機種毎にUIモジュールが存在する。608および609は制御モジュールと呼ばれ、詳細情報を取得する対象機種に特有の制御を受け持つモジュールである。UIモジュールと同様に、制御モジュールも詳細情報を表示する対象機種毎に存在する。制御Aモジュール608および制御Bモジュール609は、MIBモジュール610

を用いて管理対象デバイスからMIBデータを取得し、必要に応じてデータの変換を行い、各々対応するUIAモジュール606またはUIBモジュール607にデータを渡す。

【0042】さて、MIBモジュール610は、オブジェクト識別子とオブジェクトキーとの変換を行うモジュールである。ここでオブジェクトキーとは、オブジェクト識別子と一対一に対応する32ビットの整数のことである。オブジェクト識別子は可変長の識別子であり、ネットワーク管理ソフトウェアを実装する上で扱いが面倒なので、本願に係るネットワーク管理ソフトウェアにおいてはオブジェクト識別子と一対一に対応する固定長の識別子を内部的に用いている。MIBモジュール610より上位のモジュールはこのオブジェクトキーを用いてMIBの情報を扱う。これにより、ネットワーク管理ソフトウェアの実装が楽になる。

【0043】611はSNMPモジュールと呼ばれ、SNMPパケットの送信と受信を行う。

【0044】612は共通トランスポートモジュールと呼ばれ、SNMPデータを運搬するための下位プロトコルの差を吸収するモジュールである。実際には、動作時にユーザが選択したプロトコルによって、IPXハンドラ613かUDPハンドラ614のいずれかがデータを転送する役割を担う。なお、UDPハンドラは、実装としてWinSock617を用いている。(WinSockについては、例えばWindows socket API v1.1の仕様書を参照のこと。このドキュメントは、複数箇所から入手可能であるが、例えばマイクロソフト社製のコンパイラであるVisual C++に同梱されている。) コンフィグレータ603が用いる現在のプロトコル615というのは、動作時にユーザが選択しているIPXプロトコルかUDPプロトコルのいずれかのことを示す。

【0045】本従来例で使用する探索モジュール604とMIBモジュール610との間のインターフェースについて説明する。

【0046】MIBモジュール610は図9に示すC言語のAPI(Application Program Interface)を上位モジュールに提供する。

【0047】最初に、上位モジュールはMIBモジュールとの間で、指定したアドレスに対するインターフェース(これをポートと呼ぶ)を開設するために、MIBOpen API901呼び出しを行う。MIBモジュールは、開設されたインターフェースを識別するための識別子(これをポート識別子と呼ぶ)を上位モジュールに返す(MIBOpen API901の第一引数portに返される値)。以降上位モジュールはポート識別子を用いてMIBモジュールとのやりとりを行う。

【0048】ここで指定するアドレスは、動作しているプロトコルのアドレスであり、IPプロトコルの場合はIPアドレス、NetWareプロトコルの場合はNetWareアド

レスである。さらにブロードキャストアドレスを指定することもできる。

【0049】ブロードキャストアドレスを指定してポートをオープンした場合は、ブロードキャストアドレスに応答する複数のデバイスと通信を行うことが可能である。

【0050】上位モジュールはポートを使用しなくなつたときMIBClose API904呼び出しを行いポートを閉じる。

【0051】上位モジュールがMIBオブジェクトの読み出しが行う場合は、MIBReadObjects API902呼び出しを行なう。MIBReadObjects API902呼び出しにはポート識別子、読み出すべきMIBオブジェクトのオブジェクトキーを指定すると共に、MIBモジュールが読み出したMIBオブジェクトの値を上位へ通知するためのコールバック関数のアドレスを指定する。

【0052】MIBReadObjects API902呼び出しによりSNMPのGet-requestコマンドが生成され、ネットワーク上に送信される。図8に示したように、このGet-requestコマンドに応答するエージェントを持つデバイスはGet-responseコマンドを送信する。

【0053】上位モジュールがMIBオブジェクトへの書き込みを行なう場合は、MIBWriteObjects API903呼び出しを行なう。MIBWriteObjects API903呼び出しにはポート識別子、書き込むMIBオブジェクトのオブジェクトキーとその値を指定すると共に、MIBモジュールが書き込みの結果を上位へ通知するためのコールバック関数のアドレスを指定する。

【0054】MIBWriteObjects API903呼び出しによりSNMPのGet-requestコマンドが生成され、ネットワーク上に送信される。図8に示したように、このGet-requestコマンドに応答するエージェントを持つデバイスはGet-responseコマンドを送信する。

【0055】コールバック関数はMIBReadObjects API902もしくはMIBWriteObjects API903の結果を上位モジュールに通知するためのものである。具体的には、デバイスのアドレスと受信したGet-responseコマンドの内容を上位へ通知する。

【0056】ブロードキャストアドレスを指定してオープンされたポートに対してMIBReadObjects API902を呼び出しを行なった場合、ネットワーク上に送信されるGet-requestコマンドを運ぶパケット（IPプロトコルの場合はIPパケット、NetWareプロトコルの場合はIPXパケット）の宛先アドレスがブロードキャストアドレスになる。従って、このパケットは複数のデバイスで受信されるので、Get-requestコマンドには複数のデバイスが応答する。つまりマネージャ側では複数のGet-responseコマンドを受信

する。この場合、コールバック関数は、ポート識別子は同じでデバイスのアドレスが異なる複数回の呼び出しが行われる。上位モジュールはアドレス情報を調べることにより、そのコールバックがどのデバイスからのものかを知ることができる。

【0057】次に具体的なデータの流れを説明する。MIBモジュール610では上位からの要求によりオブジェクトキーからオブジェクトIDへの変換等の処理を行いSNMPモジュール611へコマンド送信要求を行う。SNMPモジュール611はMIBモジュール610からの送信要求によりSNMP PDUをRAM503上で組み立て、共通トランSPORTモジュール612へ送信要求を行う。共通トランSPORTモジュール612では動作プロトコルによりヘッダの付加等の所定の処理を行い、TCP/IPプロトコルであればWinSockモジュール617へ、NetWareプロトコルであればNetWare APIモジュール616へパケット送信要求を行う。以下TCP/IPプロトコルで動作しているものとして説明を行う。WinSockモジュール617は送信要求のあったパケットをIPパケット化し、OSに対してネットワークへのデータ送信要求を行う。OSはRAM503上のデータをシステムバス504を介してNIC508へ書き込む。NIC508では書き込まれたデータを所定のフレーム化してLAN100に送信する。

【0058】LAN100に接続されているデバイスからのパケットはNIC508で受信される。NIC508ではパケット受信を割り込みによりOSに通知する。OSはNIC508から受信パケットをシステムバス504経由で読み出しRAM503に置く。OSでは動作プロトコルもしくは受信したパケットからプロトコルを判断し、TCP/IPプロトコルであればWinSockモジュール617へ、NetWareプロトコルであればNetWare APIモジュール616へパケット受信が通知される。以下TCP/IPプロトコルで動作しているものとして説明を行う。WinSockモジュール617では、受信パケットが自分宛のものかどうかを受信パケット中のアドレスにより判断する。受信パケットが自分宛のものでは無いときは受信パケットを破棄する。受信パケットが自分宛であった場合は、UPDハンドラ614起動し、共通トランSPORTモジュール612にパケット受信を通知する。共通トランSPORTモジュール612ではトランSPORTヘッダの除去等の所定の処理を行い、SNMPモジュール611へパケット受信を通知する。SNMPモジュール611ではSNMPヘッダの除去等の所定の処理を行い、MIBモジュールへPDU受信を通知する。MIBモジュール610では所定処理を行うと共に受信した情報をMIB APIで規定された形式に変換し上位モジュールのコールバック関数を呼び出すことにより、デバイスからの応答を上位モジュールへ通知する。

50 【0059】なお、以下の説明において、本願に係るネ

ットワーク管理ソフトウェアのことを「NetSpot」と呼称する。

【0060】NetSpotのインストールに必要なファイルは、通常、フロッピーディスク(FD)やCD-ROMなどの物理媒体に記録されて配布されるか、あるいはネットワークを経由して伝送される。ユーザは、これらの手段によりNetSpotのインストールに必要なファイルを入手した後、所定のインストール手順に従ってNetSpotのインストールを開始する。

【0061】NetSpotのインストール手順は、他の一般的なソフトウェアのインストール手順と同様である。すなわち、ユーザがNetSpotのインストーラをパーソナルコンピュータ(PC)上で起動すると、その後はインストーラが自動的にインストールを実行する。インストーラは、NetSpotの動作に必要なファイルをPCのハードディスクにコピーし、また、必要に応じてユーザから情報を入力してもらしながら、NetSpotの動作に必要なファイルの修正または新規作成なども行う。

【0062】次に、本従来例におけるネットワーク管理プログラムにおける探索シーケンスについて説明する。

【0063】図10は、従来のネットワーク管理プログラムにおける探索シーケンスを示す図である。

【0064】図10において、探索モジュール1030はネットワーク管理プログラムの探索モジュールを示し、これは図6における604と同等である。この探索モジュールは、ネットワーク管理プログラムの他のモジュールと同様に、図1におけるPC103上で図5におけるCPU501によって実行される。

【0065】デバイス1031は、ネットワーク上に接続されており、かつSNMPエージェントが動作しているデバイスでプリンタとして標準的なMIBのみを搭載しているプリンタを示し、例えば図1におけるNB118を示す。ここで、標準的なMIBとは、RFC1213、RFC1514およびRFC1759に規定されているMIBなどを言う。

【0066】デバイス1032は、ネットワーク上に接続されており、かつSNMPエージェントが動作しているデバイスで、プリンタとして標準的なMIBおよび企業独自のプライベートMIBを搭載しているプリンタを示し、例えば図1におけるNB101を示す。本実施例では、NB101が企業独自のプライベートMIBを実装しているものとして説明を行なう。プライベートMIBは、図4の記号408で示されるcanonノード以下のMIBオブジェクトである。

【0067】上位モジュールから探索開始の指示が出されると、探索モジュールはブロードキャストアドレスを指定してデバイスの状態とデバイス種別を取得するためのGetリクエストSNMPパケットをネットワークに送出する(1001および1002)。このパケットは、ネットワークに接続されている全てのデバイスに届

けられる。

【0068】ここで、問い合わせるMIBオブジェクトとして、1001では標準的なMIBのみを、1002ではプライベートMIBを用いる。異なるMIBオブジェクトを問い合わせるのは、プライベートMIBを搭載しているデバイスと、標準MIBのみを搭載しているデバイスの両方を見つけたいためである。

【0069】このSNMPパケットに対して、SNMPエージェントを実装しているネットワークデバイスは、10それぞれ応答パケットを送出する(1003から1006)。

【0070】ここで1003は、標準MIBを問い合わせるブロードキャストパケット1001に対するデバイス1031の応答であり、デバイス1031の状態に応じた値が返る。1004は、標準MIBを問い合わせるブロードキャストパケット1001に対するデバイス1032の応答であり、デバイス1032の状態に応じた値が返る。1005は、プライベートMIBを問い合わせるブロードキャストパケット1002に対するデバイス1031の応答であり、デバイス1031はプライベートMIBを実装していないため、エラーとしてnoSuchNameが返される。1006は、プライベートMIBを問い合わせるブロードキャストパケット1002に対するデバイス1032の応答であり、デバイス1032の状態に応じた値が返る。

【0071】探索を開始してから一定の時間が経過するとい時応答タイマー1023が満了するので、探索モジュールはさらに詳細な情報を取得すべく、それぞれのデバイスに対してSNMPパケットを送信する。

【0072】より具体的には、一時探索タイマー1023が満了した時点までに、デバイス1031からはプライベートMIBに対してnoSuchNameエラーが返ってきたことがわかっているので、このデバイスをプライベートMIBには対応していないデバイス、すなわち標準MIBのみに対応しているデバイスとみなして、標準MIBを用いてより詳細な取得を行なう。この取得およびデバイス1031の返答が1007および1008である。

【0073】同様に、デバイス1032からは標準MIBおよびプライベートMIBの両方から正常な応答が返ってきてるので、このデバイスをプライベートMIBを実装しているデバイスとみなして、プライベートMIBを用いてより詳細な取得を行なう。

【0074】この取得およびデバイス1032の返答が1009および1010である。

【0075】さらに時間が経過してデバイス応答タイマー1021が満了した時点で、上位モジュールに対してそれまでに探索したデバイスの情報を通知する。

【0076】さらに時間が経過し、探索間隔タイマー1022が満了した時点で、再度探索を開始する。以降今まで説明した動作と全く同一のシーケンスであるので、

説明を省略する。

【0077】

【発明が解決しようとする課題】しかしながら、上記従来例のようにブロードキャストパケットを送信することは、そのパケットに対してNMPに応答できる全てのネットワーク機器が返信パケットを送信することになり、ネットワークのトラフィックを著しく大きくしてしまう。定期的にブロードキャストパケットを送信することはブロードキャストパケットによるトラフィック増大現象が定期的に起きることを意味し、そのような状況はネットワークの負荷および管理の観点から好ましくない。

【0078】本発明は上記従来例に鑑みてなされたもので、ブロードキャストパケットの使用を最小限に抑え、定期的なネットワークのトラフィックの増大をなくして、ネットワークの負荷を低減させることができるネットワーク管理装置及び方法を提供することを目的とする。

【0079】

【課題を解決するための手段】上記目的を達成するために、本発明は次のような構成からなる。すなわち、ブロードキャストパケットによってネットワークに接続されているデバイスを検出するためのネットワークデバイス探知手段と、前記ネットワークデバイス探知手段により検出されたデバイスを登録するためのデバイステーブルと、前記デバイステーブルに登録されたデバイスに対して、所定間隔で順次探知シーケンスを遂行する個別デバイス探知手段と、前記個別デバイス探知手段により検出された情報により、前記デバイステーブルを更新するデバイステーブル更新手段とを備える。

【0080】あるいは、前記デバイステーブルの登録内容の出力を指示された際に、前記デバイステーブルに登録された各デバイスに対して探知シーケンスを遂行し、検出された情報により前記デバイステーブルを更新する手段と、前記デバイステーブルを出力する出力手段とを更に備える。

【0081】

【発明の実施の形態】以下、添付図面に従って本発明に係る実施形態を詳細に説明する。

【0082】(第1の実施形態) 図11は、本発明におけるネットワーク管理プログラムとネットワーク上のデバイスが通信する様子を示したシーケンス図である。特に、図11ではネットワーク管理プログラムがネットワーク上に存在するデバイスを探査する様子を示している。

【0083】探索モジュール1030は本発明におけるネットワーク管理プログラムの探索モジュールを示し、これは図6における604と同等である。この探索モジュールは、本発明におけるネットワーク管理プログラムの他のモジュールと同様に、図1におけるPC103上

で図5におけるCPU501によって実行される。

【0084】1101から1107は、探索モジュール1030と、ネットワークデバイス1031および1032との間で通信が行われていることを示す。これらの通信内容については、後述のフローチャートの説明において、併に詳述する。

【0085】図12は、探索モジュール604が動作の過程において探索するデバイスの情報を記憶するために使用する検出デバイステーブルおよび前回検出テーブルと呼ばれるデータの構造を示す図である。このデータは図5におけるRAM503内に保持されている。

【0086】検出デバイステーブル1220には、検出したデバイスの情報を登録する。デバイスがひとつ検出される毎に、検出デバイステーブルの行が一行追加される。

【0087】以下、検出デバイステーブルに登録する情報について説明する。

【0088】ネットワークアドレス1201には、検出したデバイスのネットワークアドレスを登録する。ポート1202には、MIBモジュール610が提供するMibOpen関数によって返されるポート値を登録する。

【0089】デバイスタイプ1203には、デバイスの種別を登録する。

【0090】デバイス状態1204には、デバイスが現在どのような状態になっているかを登録する。「活動中」欄1205には、デバイスが現在活動中であって、ネットワークを介した通信が可能である場合にはTRUEを、そうでない場合にはFALSEを登録する。

【0091】図12の例においては、検出デバイステーブル1220には、現在ふたつのデバイスが登録されている。ひとつは、ネットワークアドレス192.1.16.130のデバイスであり、このデバイスと通信するためのMIBモジュール610におけるポート値は0x12FEである。このデバイスはデバイスタイプがレポートであり、デバイスは使用可能(Ready)であり、活動中であることを示している。また、ネットワークアドレス192.1.16.150のデバイスのポート値は0x2501であり、デバイスタイプは複写機(Copier)である。このデバイスでは現在紙詰まり(Jam)が発生しており、このデバイスもまた活動中である。

【0092】このテーブルの使用方法については、後述のフローチャートの説明において、併に詳述する。

【0093】図13は、PC103上で動作するネットワークデバイス管理プログラム全体の動作を説明したフローチャートである。

【0094】プログラムが起動されると、まずステップS1301においてブロードキャストパケットによるデバイス探索処理を行なう。ここでは、従来例の説明において図10に示したシーケンスが実行され、ブロードキ

キャストパケットによる探索処理が行われる。検出されたデバイスに関する情報は検出デバイステーブル1220に登録される。

【0095】プロードキャストパケットによる探索処理が行われるのは、この起動時のみである。

【0096】次にステップS1302に進む。ここでは、システムで発生するイベントを待ち、生じたイベントがSNMPパケットの受信であるか、デバイス応答タイマーの満了であるか、探知ポーリングタイマーの満了であるか、あるいはそれ以外のイベントであるか（全くイベントが生じていない場合も含む）を判断する。

【0097】次に、発生したイベントに応じて、それぞれ適切なサブルーチンステップS1303からステップS1306を呼び出す。

【0098】それぞれの処理が終了したら、ステップS1302に合流し、処理を繰り返す。

【0099】図14は、探知ポーリング動作を開始する際の動作を示すフローチャートである。

【0100】このシーケンスが起動される場合は二種類ある。ひとつは、例えばユーザがキーボード509あるいは不図示のマウス等の入力機器を用いて例えればメニューを選択するなどの操作によって、明示的にデバイス探知ポーリング動作を開始した場合である。この場合、図14の処理は、図13のフローチャートにおいてステップS1306のその他の処理の一部に相当する。もうひとつの場合には、図14の処理において起動する探知ポーリングタイマーが満了した場合である。この場合には、図14の処理は、図13におけるステップS1305の処理に相当する。

【0101】さて、探知ポーリング動作の開始においては、まずステップS1401において探知ポーリングの対象となるデバイスを示す変数targetに次のデバイスのデバイスのアドレスを代入する。ここで、次のデバイスとは、検出デバイステーブル1220において、前回本処理の対象となったデバイスの次のデバイスという意味である。本処理が一度も実行されていない場合には、最初の行のデバイスを示す。検出デバイステーブル1220に登録されている全デバイスに対して探知ポーリング動作が終了したかどうかを判断する。なお、検出デバイステーブル1220にデバイスがひとつも登録されていない場合には、全デバイス終了と判断する。ステップS1402において全デバイス終了と判断された場合には、処理を終了する。

【0102】一方、まだ終了していないと判断された場合には、ステップS1403に進み、探知動作を開始する。本ステップの詳細については、後程図15のフローチャートを用いて説明する。次にステップS1404に進み、一定時間後に次のデバイスを探知ポーリング動作の対象とするために探知ポーリングタイマーを起動する。探知ポーリングタイマーが満了したら、図13におけるス

テップS1305の処理、すなわち図14の処理が開始される。ステップS1404の処理が終了したら、探知ポーリング動作の処理を終了する。

【0103】図15は、探知動作開始処理を詳しく説明するフローチャートである。本処理は、図14におけるステップS1403の処理に相当する。

【0104】まずステップS1501において、探知動作がどこまで進行したかを示す探知フェーズ変数に値1を代入する。この変数は、図16で説明するパケットの

10 受信処理において使われる。次にステップS1502に進み、探知動作の対象となるデバイスのデバイス種別を示すMIBオブジェクトに対して、図9で説明したMibReadObject関数を呼び出すことによってSNMPのGetRequestを発行してデバイス種別を要求する。次にステップS1503でデバイス応答タイマーを起動する。デバイスから応答がなく、デバイス応答タイマーが満了すると図15のステップS1304の処理が起動される。ステップS1503でデバイス応答タイマーを起動したら、探知動作開始処理を終了する。

【0105】図16は、パケット受信処理を詳しく説明するフローチャートである。本処理は、図13におけるステップS1303に相当する。パケット受信処理においては、まずステップS1601において、現在探知対象になっているデバイスからのパケット受信であるかどうかを判断する。この判断は、受信したパケットの送信元アドレスが探知対象のデバイスのものであるか、さらにはSNMPのリクエストIDが探知対象のデバイスに送信した値と同じであったかどうかを判断することによ

30 って行われる。ステップS1601において、探知対象のデバイスからのパケット受信であると判断された場合には、ステップS1602に進み、探知フェーズが1であるかどうかを判断する。探知フェーズが1である場合には、ステップS1603に進み、デバイスから応答があったので図15のステップS1503で起動したデバイス応答タイマーを削除する。

【0106】次にステップS1604に進み、受信したパケットには図15のステップS1502で要求したデバイス種別が格納されているはずなので、その値を取り出すとともに、検出デバイステーブル1220のデバイスタイプ1203の該当する行に取得したデバイスタイプを上書きする。ここで、既存の値を上書きするのは、同じアドレスに対してデバイスが入れ替えられて別のデバイスが設置されている場合があるので、最新の情報を反映するためである。次に、ステップS1605では、探知シーケンスの進み具合を示す探知フェーズ変数に値2を代入する。次にステップS1606に進み、MibReadObject関数を用いてデバイスに対してデバイス状態を示すMIBオブジェクトの値を要求す

【0107】さて、ステップS1602において探知フェーズが1ではないと判断された場合、探知フェーズはデバイス状態を要求したフェーズ2であるのでステップS1607に進み、受信パケットの中からデバイス状態を示す値を取り出して、検出デバイステーブル1220のデバイス状態1204の該当する行に取り出した値を格納する。次にステップS1608に進み、探知対象のデバイスから必要な値全てが取り出すことが出来てデバイスが活動中であることが確認できたので、デバイステーブル1220の活動中欄1205の該当する行に活動中を示す値であるTRUEを書き込む。

【0108】最後に図13のステップS1304において、デバイス応答タイマーが満了した場合の処理について説明する。デバイス応答タイマーが満了するのは、図15のステップS1502におけるデバイスへのデバイス種別値の要求に対してデバイスが応答を返さなかった場合である。このような場合、デバイスは電源が切られているか、あるいは既にネットワークから切り離された状態であると考えられるため、図12の検出デバイステーブル1220の活動中欄1205の該当行に活動中ではないことを示す値FALSEを登録するとともに、CRTC506を操作してCRTにその旨を表示し、ユーザに通知する。

【0109】ユーザへの表示の様子を図18に示す。ウィンドウ1801は、検出デバイステーブルに登録されているデバイスが表示されており、それらのうち探知シーケンスによって活動中ではないと判断されたデバイスは、×印1803がつけられている。

【0110】以上説明したように、本実施例によれば、定期的にブロードキャストパケットを送信することによりデバイステーブルを更新する代わりに、デバイステーブルに載っているデバイスに対して順番にその状態を検出するためのパケットを送信して、デバイスが反応するかどうかだけを調べることができる。これにより比較的少ないネットワークトラフィックでデバイステーブルを更新することができる。

【0111】(第2の実施形態) 第1の実施例によると、探索におけるブロードキャストパケットの送信が最初の一回だけに限定され、その後はデバイス毎に比較的長い時間間隔で、しかも一度にひとつのデバイスに対して探知シーケンスが起動されるため、ネットワークのトラフィックが一時的に増大するようなことはない。

【0112】しかしながら、ひとつあたりのデバイスを考えた場合、探知シーケンスが起動されてから、次同じデバイスに対して探知シーケンスが実行されるまでの間隔は、ブロードキャストを用いていた従来例に比較して長くなっている。このような状況下では、古いネットワークプリンタをネットワークから切り離して、代わりに新しいプリンタを設置したような場合には、検出デバイステーブルに登録されている機種情報が実際に設置され

ているデバイスと異なる可能性が高い。

【0113】そのような場合にデバイス詳細ウィンドウ1802を開いても、デバイス種別が異なるために、正しい情報が得られない。

【0114】そこで本発明の第2の実施例では、デバイス詳細ウィンドウを開く前には必ず探知シーケンスを実行することにより、検出デバイステーブルの情報を最新の情報に更新するとともに必ず正しくデバイス詳細ウィンドウを開けるようにしたものである。

10 【0115】以下、フローチャートを用いて説明する。

【0116】図17は、本発明の第2の実施例の動作を説明するフローチャートである。第2の実施例において、このフローチャート以外の構成については基本的に第1の実施例と同じである。

【0117】図17で示されるデバイス詳細ウィンドウを開く手順は、ユーザがメニューからデバイス詳細ウィンドウを開く操作を行なった時に起動され、第一の実施例における図13のステップS1306のその他の処理の一部となる。

20 【0118】本手順では、まずステップS1701においてユーザがデバイス詳細ウィンドウを開こうとしたデバイスに対して探知シーケンスを起動する。このシーケンスは、第1の実施例においてデバイス一台あたりの情報を取得するまでに相当する。次にステップS1702において探知された情報と検出デバイステーブル1220の情報を比較して、それらが同じ内容であるかどうかを調べる。もし同じ情報であればステップS1703に進み、デバイス詳細ウィンドウ1802を開き、デバイスから詳細な情報を取得してその内容をウィンドウ1802に表示する。一方ステップS1702において異なる情報であると判断された場合にはステップS1704に進み、検出デバイステーブル1220の情報を更新する。次にステップS1705に進み、例えばポップアップウィンドウを表示するなどの方法によってユーザにデバイスが入れ替えられている旨を通知する。次にステップS1703の処理に合流する。

30 【0119】以上のように、本実施例によれば、デバイス詳細ウィンドウを開くときに、デバイスとネットワークボードの種別を確かめる探知シーケンスを必ず実行することができる。これにより、デバイスの探索の頻度が下がっても、デバイス参照ウィンドウには、それが開かれた際に最新の状態を確実に表示することができる。このため、探知間隔よりも短いタイミングでデバイスあるいはネットワークボードが異なる種別のものに入れ替えていたときであっても正しくデバイス詳細ウィンドウを開くことができる。

40 【0120】上記で説明した本発明に係るネットワークデバイス制御プログラムは、外部からインストールされるプログラムによって、PC500によって遂行されており、その場合、そのプログラムはCD-ROMやフ

ラッシュメモリやフロッピーディスクなどの記憶媒体により、あるいは電子メールやパソコン通信などのネットワークを介して、外部の記憶媒体からプログラムを含む情報群をPC500上にロードすることにより、PC500に供給される場合でも本発明は適用されるものである。

【0121】図19は、記憶媒体の一例であるCD-R ROMのメモリマップを示す図である。9999はディレクトリ情報を記憶してある領域で、以降のインストールプログラムを記憶してある領域9998およびネットワークデバイス制御プログラムを記憶してある領域9997の位置を示している。9998は、インストールプログラムを記憶してある領域である。9997は、ネットワークデバイス制御プログラムを記憶してある領域である。本発明のネットワーク制御プログラムがPC500にインストールされる際には、まずインストールプログラムを記憶してある領域9998に記憶されているインストールプログラムがシステムにロードされ、CPU501によって実行される。次に、CPU501によって実行されるインストールプログラムが、ネットワークデバイス制御プログラムを記憶してある領域9997からネットワークデバイス制御プログラムを読み出して、ハードディスク511に格納する。

【0122】なお、本発明は、複数の機器（例えばホストコンピュータ、インターフェース機器、リーダなど）から構成されるシステムあるいは統合装置に適用しても、ひとつの機器からなる装置に適用してもよい。

【0123】また、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはCPUやMPU）が記憶媒体に格納されたプログラムコードを読み出し実行することによっても、本発明の目的が達成されることは言うまでもない。

【0124】この場合、記憶媒体から読み出されたプログラムコード自体が本発明の新規な機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。

【0125】プログラムコードを供給するための記憶媒体としては、例えば、フロッピーディスク、ハードディスク、光ディスク、光磁気ディスク、CD-ROM、CD-R、磁気テープ、不揮発性のメモリカード、ROMなどを用いることができる。

【0126】また、コンピュータが読み出したプログラムコードを実行することによって、前述した実施形態の機能が実現される他、そのプログラムコードの指示に基づき、コンピュータ上で稼動しているOSなどが実際の処理の一部または全部を行い、その処理によっても前述した実施形態の機能が実現され得る。

【0127】さらに、記憶媒体から読み出されたプログ

ラムコードが、コンピュータに挿入された機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに書き込まれた後、そのプログラムコードの指示に基づき、その機能拡張ボードや機能拡張ユニットに備わるCPUなどが実際の処理の一部または全部を行い、その処理によっても前述した実施形態の機能が実現され得る。

【0128】なお、本発明は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した

10 記憶媒体から、そのプログラムをパソコン通信など通信ラインを介して要求者にそのプログラムを配信する場合にも適用できることは言うまでもない。

【0129】

【発明の効果】以上説明したように、本発明の第一の実施例によれば、定期的にブロードキャストパケットを送信することによりデバイステーブルを更新する代わりに、デバイステーブルに載っているデバイスに順番にパケットを送信して、デバイスが反応するかどうかだけを調べることができる。これにより比較的少ないネットワ

20 ークトラフィックでデバイステーブルを更新することができる。

【0130】さらに本発明の第二の実施例によれば、デバイス詳細ウインドウを開くときに、デバイスとネットワークボードの種別を確かめる探知シーケンスを必ず実行することができる。これにより、探知間隔よりも短いタイミングでデバイスあるいはネットワークボードが異なる種別のものに入れ替えられていたときであっても正しくデバイス詳細ウインドウを開くことができる。

【図面の簡単な説明】

30 【図1】プリンタをネットワークに接続するたあのネットワークボードを、開放型アーキテクチャを持つプリンターへつなげた場合を示す図である。

【図2】エージェントを実装したネットワークボードをプリンタに接続する実施形態を示す断面図である。

【図3】ネットワークボードとプリンターとLANとの電気的接続を示すブロック図である。

【図4】MIBの構造を示す概念図である。

【図5】ネットワーク管理ソフトウェアが稼動可能なPCの構成を示すブロック図である。

40 【図6】ネットワーク管理ソフトウェアのモジュール構成図である。

【図7】SNMPメッセージのフォーマットを示す図である。

【図8】マネージャとエージェント間でのSNMPコマンドのやりとりを示す図である。

【図9】MIBモジュールAPIを示す図である。

【図10】従来例における探索モジュールとデバイス間の通信シーケンスを示す図である。

【図11】本発明の実施例における探索モジュールとデバイス間の通信シーケンスを示す図である。

50 バイス間の通信シーケンスを示す図である。

【図12】検出デバイステーブルの構造を示す図である。

【図13】ネットワークデバイス管理プログラム全体の動作を説明したフローチャートである。

【図14】探知ポーリング動作を開始する際の動作を示すフローチャートである。

【図15】探知動作開始処理を詳しく説明するフローチャートである。

【図16】パケット受信処理を詳しく説明するフローチャートである。

【図17】本発明の第2の実施例の動作を説明するフロ

ーチャートである。

【図18】ユーザへの画面表示の方法を一例を示す図である。

【図19】本発明のネットワーク管理ソフトウェアの記憶媒体におけるメモリマップを示す図である。

【符号の説明】

S 1301 ブロードキャストパケット送信手段

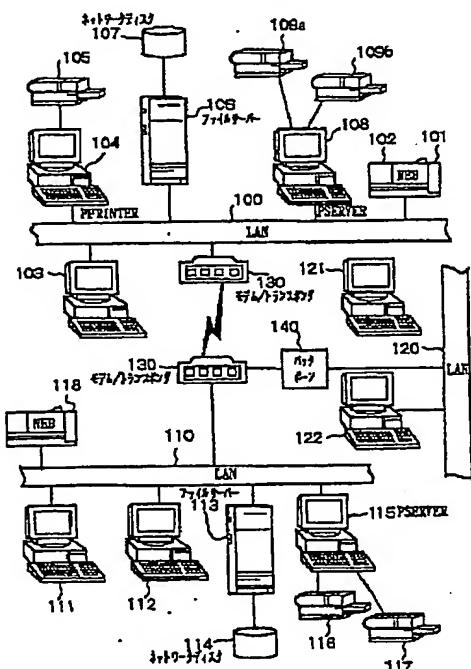
S 1220 検出デバイステーブル

S 1404 探知ポーリングタイマー起動手段

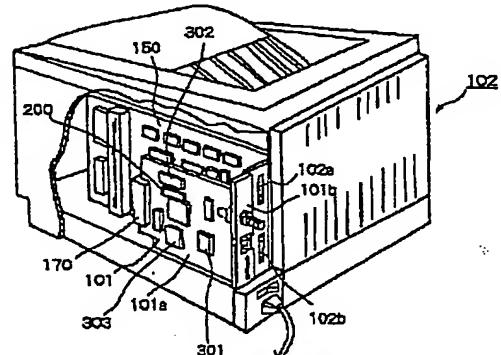
10 S 1403 探知シーケンス開始手段

S 1608 検出デバイステーブル更新手段

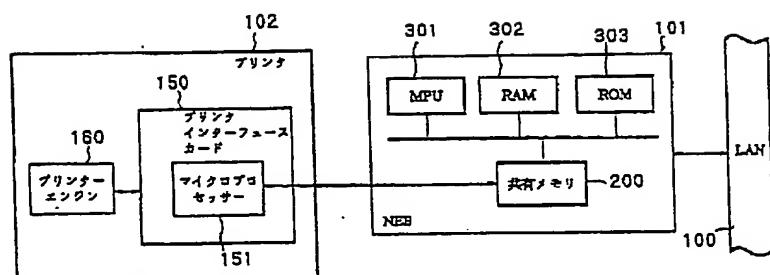
【図1】



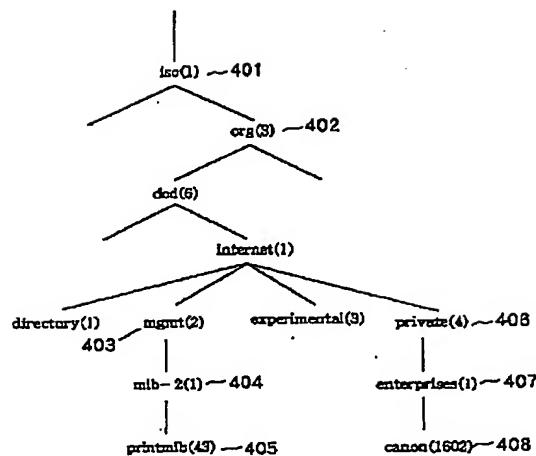
【図2】



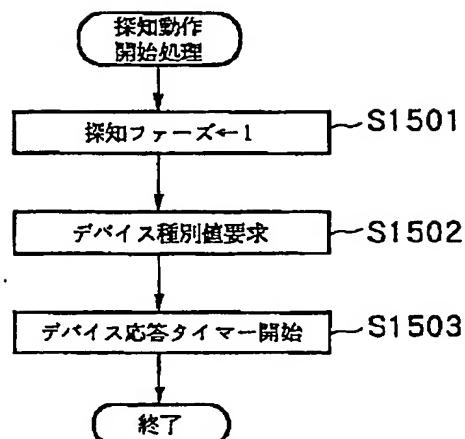
【図3】



【図4】

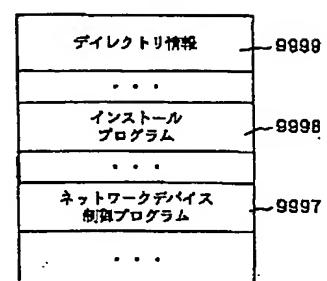
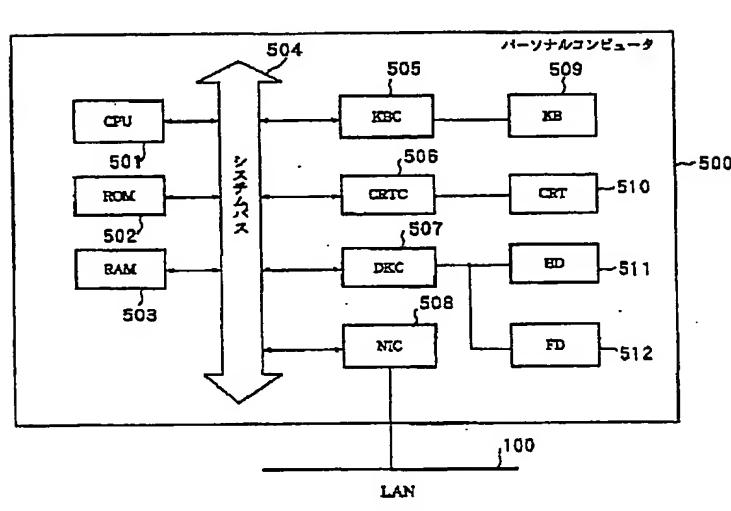


【図15】

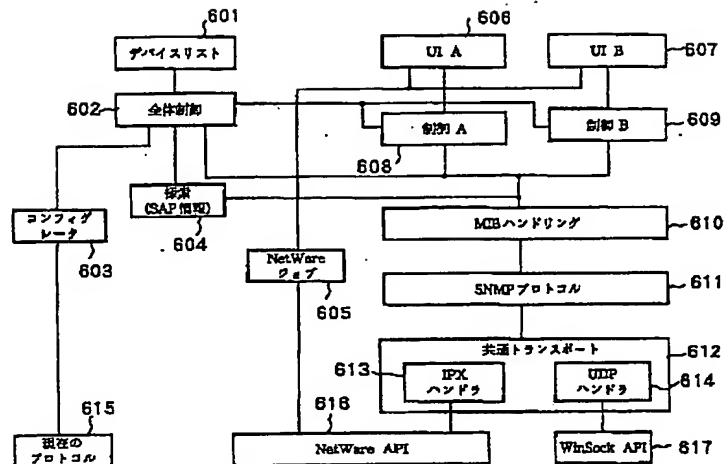


【図19】

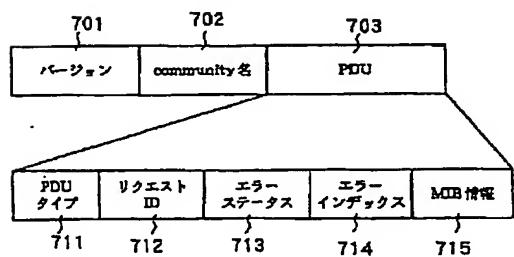
【図5】



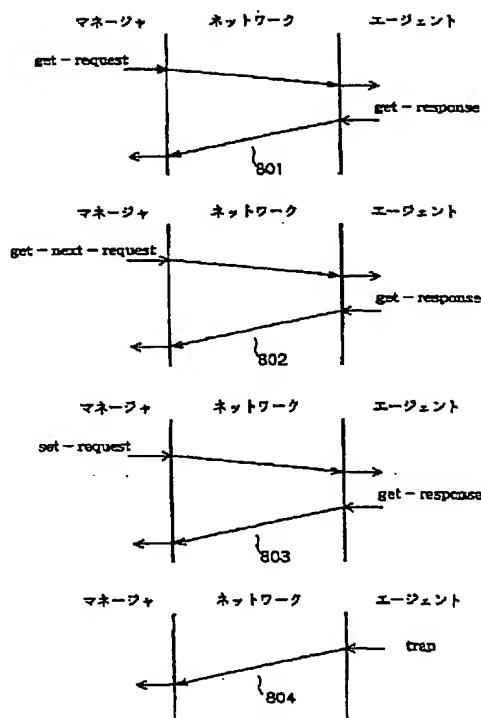
【図6】



【図7】



【図8】



【図9】

```

901
BOOL MIBOpen( int *port, ADDR ADDR *addr ):

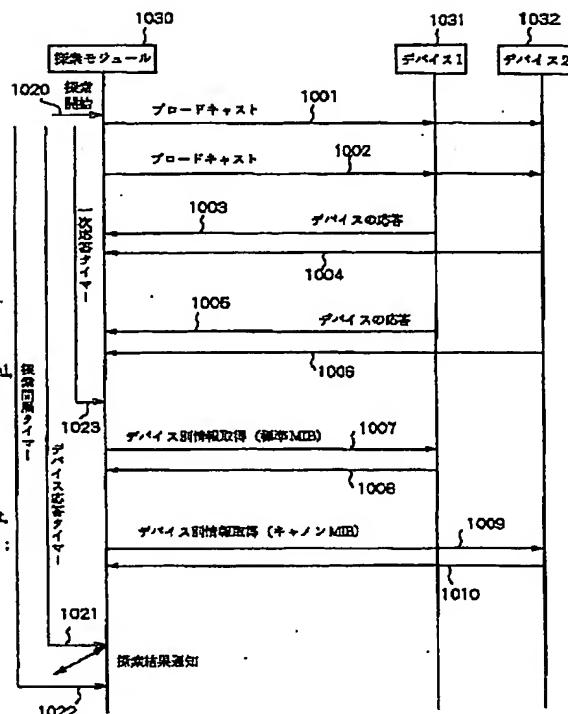
902
BOOL MIBReadObjects( int port, int count, MIBOBJ *obj,
CALLBACK respproc ):

903
BOOL MIBWriteObjects( int port, int count, MIBOBJVAL *objval,
CALLBACK respproc ):

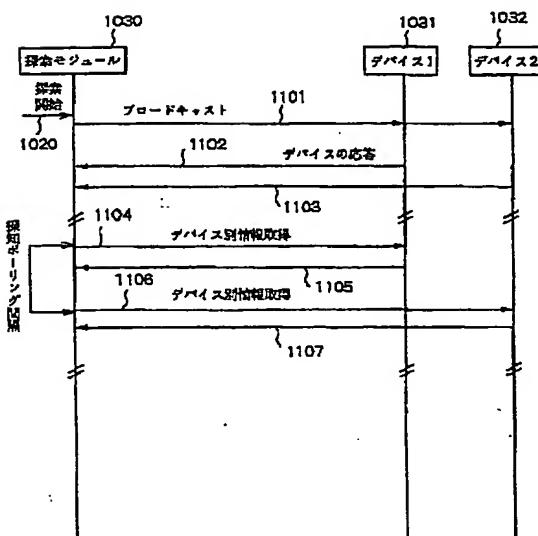
904
BOOL MIBClose( int port );

905
typedef VOID (*CALLBACK)( int port, ADDR *addr, INT result,
int count, MIBOBJVAL *objval );

```



【図11】

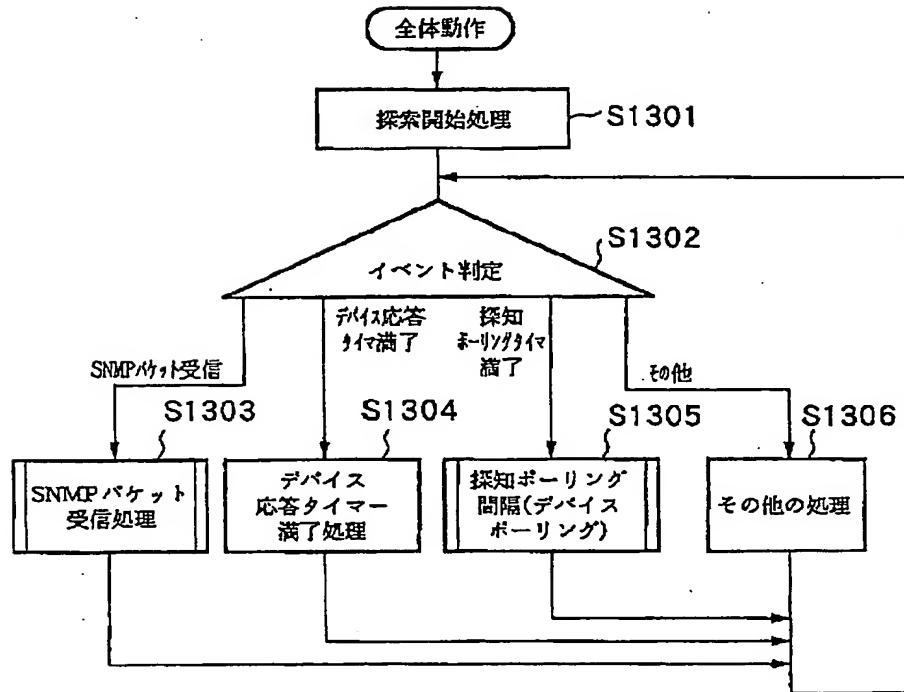


【図12】

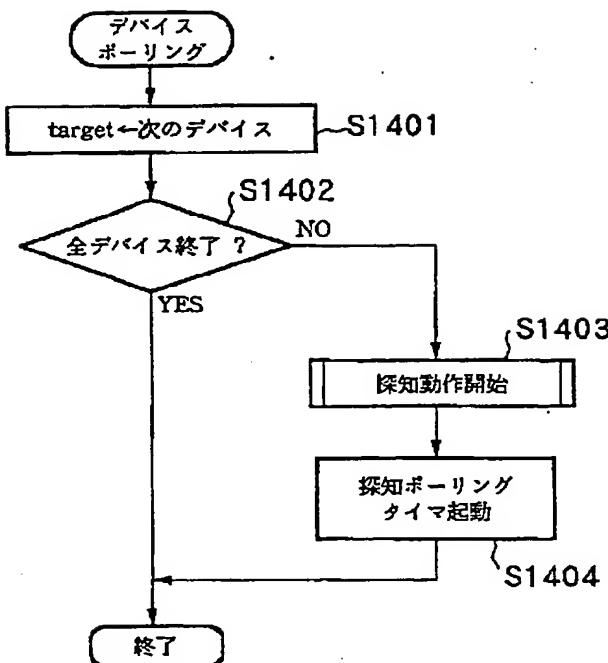
【図12】

検出デバイステーブル1220				
ネットワークアドレス	ポート	デバイスタイプ	デバイス名	活動中
192.168.16.130	0x12FE	LBP	Ready	TRUE
192.168.16.150	0x2501	Copier	Jam	TRUE
-	-	-	-	-

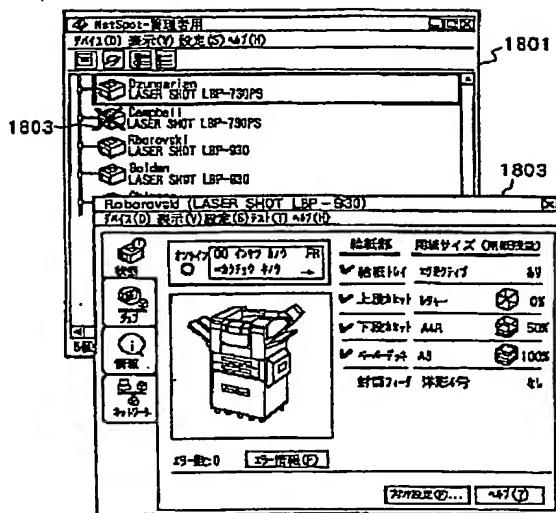
【図13】



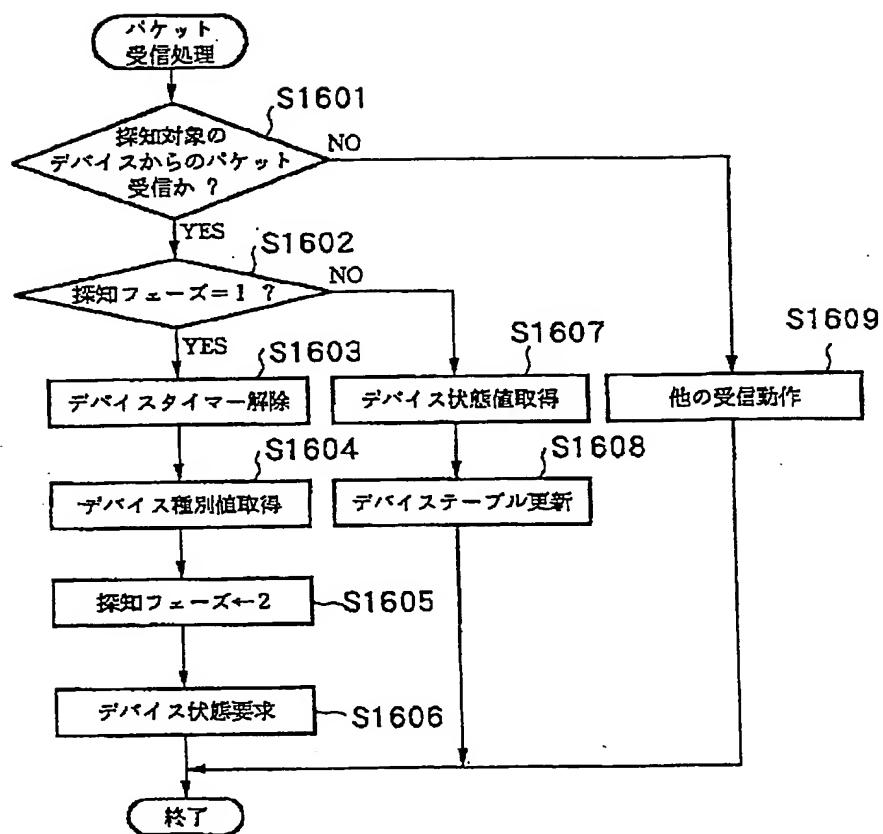
【図14】



【図18】



【図16】



【図17】

